**1.Write a for loop that prints the even numbers from 1 to 20.**

```
for (int i = 1; i <= 20; i++) {
if (i % 2 == 0) {
System.out.println(i);
}
}
```

---

**2.Create a while loop that prompts the user for their flight choice until a valid number is entered.**

```
import java.util.Scanner;

public class FlightChoice {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
int flightNumber = -1;

while (flightNumber < 0) {
System.out.println("Enter a valid flight number: ");
if (scanner.hasNextInt()) {
flightNumber = scanner.nextInt();
} else {
System.out.println("Invalid input. Please enter a number.");
scanner.next(); // Clear invalid input
}
}

System.out.println("You selected flight number: " + flightNumber);
}
}
```

---

**3.Discuss the pros and cons of using different types of loops for iterating through an array of numbers.**

Pros of for loop:

- Clear and concise for fixed iterations.
- Easily integrates with indices for arrays.

Cons of for loop:

- Less flexible for conditions outside array bounds.

Pros of while loop:

- Flexible for dynamic conditions.

Cons of while loop:

- Risk of infinite loops if conditions are mishandled.

---

**4.Write a Java program that uses a for loop to print the first 10 numbers of the Fibonacci sequence.**

```
public class Fibonacci {
public static void main(String[] args) {
int a = 0, b = 1;
System.out.println(a);
```

```
System.out.println(b);
for (int i = 2; i < 10; i++) {
int next = a + b;
System.out.println(next);
a = b;
b = next;
}
}
}
```

---

**5.Create a Java program using a while loop to calculate the sum of integers from 1 to 100.**

```
public class SumCalculator {
public static void main(String[] args) {
int sum = 0;
int i = 1;
while (i <= 100) {
sum += i;
i++;
}
System.out.println("Sum: " + sum);
}
}
```

---

**6.Implement a do-while loop that prompts the user to enter a number until they enter a negative number.**

```
import java.util.Scanner;

public class NegativeInput {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
int number;

do {
System.out.println("Enter a number: ");
number = scanner.nextInt();
} while (number >= 0);

System.out.println("Negative number entered. Exiting.");
}
}
```

---

**7.Write a Java program that demonstrates the use of the continue statement in a loop.**

```
public class ContinueDemo {
public static void main(String[] args) {
for (int i = 1; i <= 10; i++) {
if (i % 2 == 0) {
continue;
}
System.out.println(i);
}
}
}
```

**8.Initialize and print a 2D array of integers in Java.**

```
public class TwoDArray {
public static void main(String[] args) {
int[][] array = {
{1, 2, 3},
{4, 5, 6},
{7, 8, 9}
};

for (int i = 0; i < array.length; i++) {
for (int j = 0; j < array[i].length; j++) {
System.out.print(array[i][j] + " ");
}
System.out.println();
}
}
}
```

---

**9.Compare and contrast the use of for loops and while loops. When would you prefer one over the other?**

for loops:

- Best for known iteration counts.
- Compact and clear syntax for counter-driven logic.

while loops:

- Best for dynamic conditions or indeterminate iteration counts.

---

**10.Analyze the impact of using a break statement in nested loops. What considerations should be made?**

Using break in nested loops:

- Exits only the innermost loop.
- For outer loop exits, label the loop and use break with the label.

Considerations:

- May reduce code readability.
- Alternative logic can simplify understanding.

---

**11.How do arrays improve the organization and management of data in Java? Discuss with examples.**

Arrays provide structured data storage, allowing access by index. Example:

```
int[] numbers = {1, 2, 3, 4, 5};
for (int number : numbers)
{
System.out.println(number);
}
```

**12.Examine the differences in memory allocation for single-dimensional and multi-dimensional arrays.**

Single-dimensional arrays:

- Contiguous memory allocation.

Multi-dimensional arrays:

- Row-by-row memory allocation.
- Higher memory overhead.

---

**13.Discuss the potential pitfalls of using uninitialized arrays in Java.**

- Uninitialized arrays cause NullPointerException when accessed.
- Ensure proper initialization to avoid runtime errors.

---

**14.Create a method that accepts an array and returns the maximum value using a for loop.**

```
public static int findMax(int[] array) {
int max = array[0];
for (int i = 1; i < array.length; i++) {
if (array[i] > max) {
max = array[i];
}
}
return max;
}
```

---

**15. Write a Java program that finds the average of numbers stored in an integer array.**

```
public class ArrayAverage {
public static void main(String[] args) {
int[] numbers = {2, 4, 6, 8, 10};
int sum = 0;
for (int number : numbers) {
sum += number;
}
double average = (double) sum / numbers.length;
System.out.println("Average: " + average);
}
}
```

---

**16. Write a Java program that sums the elements of a 2D array.**

```
public class TwoDArraySum {
public static void main(String[] args) {
int[][] array = {
{1, 2, 3},
{4, 5, 6},
{7, 8, 9}
};
int sum = 0;
for (int i = 0; i < array.length; i++) {
```

```
for (int j = 0; j < array[i].length; j++) {
sum += array[i][j];
}
}
System.out.println("Sum: " + sum);
}
}
```

---

**17. Demonstrate how to find the minimum and maximum values in a given array.**

```
public class MinMaxFinder {
public static void main(String[] args) {
int[] numbers = {5, 2, 8, 1, 9};
int min = numbers[0];
int max = numbers[0];

for (int number : numbers) {
if (number < min) {
min = number;
}
if (number > max) {
max = number;
}
}

System.out.println("Minimum: " + min);
System.out.println("Maximum: " + max);
}
}
```

---

**18. Discuss the benefits and drawbacks of using static arrays versus dynamic arrays in Java.**

Static arrays:

- Fixed size, better performance.
- Limited flexibility.

Dynamic arrays:

- Flexible size (e.g., ArrayList).
- Higher memory overhead.

---

**19. Design a Java program that merges two sorted arrays into a single sorted array.**

```
import java.util.Arrays;

public class MergeSortedArrays {
public static void main(String[] args) {
int[] array1 = {1, 3, 5};
int[] array2 = {2, 4, 6};
int[] mergedArray = new int[array1.length + array2.length];

int i = 0, j = 0, k = 0;

while (i < array1.length && j < array2.length) {
if (array1[i] < array2[j]) {
mergedArray[k++] = array1[i++];
```

```
} else {
mergedArray[k++] = array2[j++];
}
}

while (i < array1.length) {
mergedArray[k++] = array1[i++];
}

while (j < array2.length) {
mergedArray[k++] = array2[j++];
}

System.out.println(Arrays.toString(mergedArray));
}
}
```

---

**20. Write a Java program to reverse the array.**

```
public class ReverseArray {
public static void main(String[] args) {
int[] array = {1, 2, 3, 4, 5};
for (int i = 0; i < array.length / 2; i++) {
int temp = array[i];
array[i] = array[array.length - 1 - i];
array[array.length - 1 - i] = temp;
}
for (int num : array) {
System.out.print(num + " ");
}
}
}
```

---

**21. Find the second largest element in Java.**

```
public class SecondLargest {
public static void main(String[] args) {
int[] numbers = {5, 1, 9, 3, 8};
int first = Integer.MIN_VALUE;
int second = Integer.MIN_VALUE;

for (int number : numbers) {
if (number > first) {
second = first;
first = number;
} else if (number > second && number != first) {
second = number;
}
}

System.out.println("Second largest: " + second);
}
}
```

**22. Find the first even number in a list and break the loop when it finds.**

```
public class FindFirstEven {
public static void main(String[] args) {
int[] numbers = {1, 3, 5, 4, 7};
for (int number : numbers) {
if (number % 2 == 0) {
System.out.println("First even number: " + number);
break;
}
}
}
}
```

---

**23. Prints all odd numbers from 1 to 20, using continue to skip even numbers.**

```
public class PrintOdds {
public static void main(String[] args) {
for (int i = 1; i <= 20; i++) {
if (i % 2 == 0) {
continue;
}
System.out.println(i);
}
}
}
```

---

**24. Prompts the user to enter numbers until they enter a negative number.**

```
import java.util.Scanner;

public class PromptUntilNegative {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
int number;

do {
System.out.println("Enter a number: ");
number = scanner.nextInt();
} while (number >= 0);

System.out.println("Negative number entered. Exiting.");
}
}
```

---

**25. Prints a multiplication table but skips the multiplication by 5.**

```
public class SkipFiveMultiplication {
public static void main(String[] args) {
for (int i = 1; i <= 10; i++) {
if (i == 5) {
continue;
}
System.out.println("5 x " + i + " = " + (5 * i));
}
}}
```

**26. Program counts from 1 to 10 but breaks when it reaches 6.**

```
public class BreakAtSix {
public static void main(String[] args) {
for (int i = 1; i <= 10; i++) {
if (i == 6) {
break;
}
System.out.println(i);
}
}
}
```

---

**27. Program prints numbers from 1 to 10 but skips the number 5.**

```
public class SkipFive {
public static void main(String[] args) {
for (int i = 1; i <= 10; i++) {
if (i == 5) {
continue;
}
System.out.println(i);
}
}
}
```

---

**28. Develop a program that checks whether a given number is prime or not. Use a for loop to test divisibility. If the number is found to be divisible by any number other than 1 and itself, it is not prime.**

```
import java.util.Scanner;

public class PrimeCheck {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Enter a number: ");
int number = scanner.nextInt();
boolean isPrime = true;

if (number <= 1) {
isPrime = false;
} else {
for (int i = 2; i <= number / 2; i++) {
if (number % i == 0) {
isPrime = false;
break;
}
}
}
System.out.println("Is the number prime? " + isPrime);
}
}
```

**29. Create a program that reverses the digits of a given integer. Use a while loop to extract each digit and build the reversed number.**

```
import java.util.Scanner;

public class ReverseDigits {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("Enter a number: ");
int number = scanner.nextInt();
int reversed = 0;

while (number != 0) {
int digit = number % 10;
reversed = reversed * 10 + digit;
number /= 10;
}

System.out.println("Reversed number: " + reversed);
}
}
```

---

**30. Write a program that prints the multiplication table for a given number. The user should input the number and the range (e.g., up to 10 or 20). Use a for loop to generate the table.**

```
import java.util.Scanner;

public class MultiplicationTable {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);

System.out.print("Enter the number: ");
int number = scanner.nextInt();

System.out.print("Enter the range: ");
int range = scanner.nextInt();

for (int i = 1; i <= range; i++) {
System.out.println(number + " x " + i + " = " + (number * i));
}
}
}
```

---

**31. Write a program that counts the number of vowels and consonants in a given string. Use a for loop to iterate through the string and keep track of the counts.**

```
import java.util.Scanner;

public class VowelConsonantCounter {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);

System.out.print("Enter a string: ");
String input = scanner.nextLine().toLowerCase();

int vowels = 0, consonants = 0;

for (char ch : input.toCharArray()) {
if (ch >= 'a' && ch <= 'z') {
if ("aeiou".indexOf(ch) != -1) {
```

```
vowels++;
} else {
consonants++;
}
}
}

System.out.println("Vowels: " + vowels);
System.out.println("Consonants: " + consonants);
}
}
```

---

**32. Print the pattern as given:**

```
1 1 1 1 1
1 1 1 1
1 1 1
1 1
1
public class PatternPrinter {
public static void main(String[] args) {
for (int i = 5; i >= 1; i--) {
for (int j = 1; j <= i; j++) {
System.out.print("1 ");
}
System.out.println();
}
}
}
```

---

**33. Feedback collection system with average rating.**

```
import java.util.Scanner;

public class FeedbackCollection {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
int totalRating = 0, count = 0;

while (true) {
System.out.print("Enter your rating (1-5, or 0 to stop): ");
int rating = scanner.nextInt();

if (rating == 0) break;
if (rating >= 1 && rating <= 5) {
totalRating += rating;
count++;
} else {
System.out.println("Invalid rating, try again.");
}
}

if (count > 0) {
System.out.println("Average Rating: " + (totalRating / (double) count));
System.out.println("Number of Ratings: " + count);
} else {
System.out.println("No ratings received.");
}
}
}
```

**34. Track monthly expenses.**

```
import java.util.Scanner;

public class MonthlyExpenses {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
double totalExpenses = 0.0;

while (true) {
System.out.print("Enter expense category and amount (or type 'done'): ");
String input = scanner.nextLine();

if (input.equalsIgnoreCase("done")) break;

String[] parts = input.split(" ");
if (parts.length == 2) {
try {
double amount = Double.parseDouble(parts[1]);
totalExpenses += amount;
} catch (NumberFormatException e) {
System.out.println("Invalid amount, try again.");
}
} else {
System.out.println("Invalid format, try again.");
}
}

System.out.println("Total Expenses: Rs. " + totalExpenses);
}
}
```

---

**35. Password validation system.**

```
import java.util.Scanner;

public class PasswordValidator {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);

while (true) {
System.out.print("Create a password: ");
String password = scanner.nextLine();

if (password.length() >= 8 && password.matches(".*[!@#$%^&*()].*") && password.matches(".*\\d.*")) {
System.out.println("Password successfully created.");
break;
} else {
System.out.println("Password must be at least 8 characters long, contain a special character, and a number.");
}
}
}
}
```

---

**36. Fitness app for logging daily steps.**

```
import java.util.Scanner;

public class FitnessApp {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
int[] steps = new int[7];
```

```
for (int i = 0; i < 7; i++) {
System.out.print("Enter steps for day " + (i + 1) + ": ");
steps[i] = scanner.nextInt();
}

int totalSteps = 0;
for (int step : steps) {
totalSteps += step;
}

System.out.println("Total steps: " + totalSteps);
System.out.println("Average steps: " + (totalSteps / 7.0));
}
}
```

---

**37. Temperature conversion tool.**

```
import java.util.Scanner;

public class TemperatureConverter {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);

while (true) {
System.out.print("Enter temperature (C or F) or type 'exit': ");
String input = scanner.nextLine();

if (input.equalsIgnoreCase("exit")) break;

try {
char scale = input.charAt(input.length() - 1);
double value = Double.parseDouble(input.substring(0, input.length() - 1));

if (scale == 'C' || scale == 'c') {
System.out.println("Fahrenheit: " + (value * 9 / 5 + 32));
} else if (scale == 'F' || scale == 'f') {
System.out.println("Celsius: " + ((value - 32) * 5 / 9));
} else {
System.out.println("Invalid scale, use 'C' or 'F'.");
}
} catch (NumberFormatException e) {
System.out.println("Invalid input format.");
}
}
}
}
```

---

**38. Simple banking system.**

```
import java.util.ArrayList;
import java.util.Scanner;

public class BankingSystem {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
double balance = 0.0;
ArrayList<String> transactions = new ArrayList<>();

while (true) {
System.out.print("Enter transaction (deposit/withdraw/exit): ");
```

```
String action = scanner.next();

if (action.equalsIgnoreCase("exit")) break;

System.out.print("Enter amount: ");
double amount = scanner.nextDouble();

if (action.equalsIgnoreCase("deposit")) {
balance += amount;
transactions.add("Deposited Rs. " + amount);
} else if (action.equalsIgnoreCase("withdraw")) {
if (amount <= balance) {
balance -= amount;
transactions.add("Withdrew Rs. " + amount);
} else {
System.out.println("Insufficient funds.");
}
} else {
System.out.println("Invalid transaction.");
}
}

System.out.println("Final Balance: Rs. " + balance);
System.out.println("Transaction History: " + transactions);
}
}
```

---

**39. Program to input grades and calculate statistics.**

```
import java.util.Scanner;

public class GradeStatistics {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
int highestGrade = Integer.MIN_VALUE;
int totalGrades = 0, count = 0, passedCount = 0;
final int PASS_THRESHOLD = 40;

while (true) {
System.out.print("Enter grade (-1 to stop): ");
int grade = scanner.nextInt();

if (grade == -1) break;

if (grade >= 0 && grade <= 100) {
totalGrades += grade;
count++;
if (grade > highestGrade) highestGrade = grade;
if (grade >= PASS_THRESHOLD) passedCount++;
} else {
System.out.println("Invalid grade, try again.");
}
}

if (count > 0) {
System.out.println("Average Grade: " + (totalGrades / (double) count));
System.out.println("Highest Grade: " + highestGrade);
System.out.println("Number of Passed Students: " + passedCount);
} else {
System.out.println("No grades entered.");
```

```
}
}
}
```

---

**40. Shopping cart application.**

```
import java.util.ArrayList;
import java.util.Scanner;

public class ShoppingCart {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
ArrayList<String> items = new ArrayList<>();
double totalPrice = 0.0;

while (true) {
System.out.print("Enter item name (or type 'checkout'): ");
String itemName = scanner.nextLine();

if (itemName.equalsIgnoreCase("checkout")) break;

System.out.print("Enter price for " + itemName + ": ");
double price = scanner.nextDouble();
scanner.nextLine(); // Consume newline

items.add(itemName + " - Rs. " + price);
totalPrice += price;
}

System.out.println("Items Purchased:");
for (String item : items) {
System.out.println(item);
}

System.out.println("Total Amount Due: Rs. " + totalPrice);
}
}
```

---

**41: Write a program that calculates the total sales and commission for a group of salespeople. Prompt the user to enter sales figures for each salesperson in a loop. The loop should continue until a negative number is entered, indicating the end of input. Calculate and display the total sales and the average sales per salesperson.**

```
import java.util.Scanner;

public class SalesAndCommission {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      double totalSales = 0;
      int salesCount = 0;

      System.out.println("Enter sales figures for each salesperson (negative number to stop):");
      while (true) {
         double sales = scanner.nextDouble();
         if (sales < 0) {
            break;
         }
         totalSales += sales;
         salesCount++;
      }
```

```
        double averageSales = salesCount > 0 ? totalSales / salesCount : 0;
        System.out.println("Total Sales: " + totalSales);
        System.out.println("Average Sales per Salesperson: " + averageSales);
    }
}
```

---

**42: Write a Java program to reverse a String.**

```
import java.util.Scanner;

public class ReverseString {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string to reverse:");
        String input = scanner.nextLine();
        String reversed = new StringBuilder(input).reverse().toString();
        System.out.println("Reversed String: " + reversed);
    }
}
```

---

**43: How would you check if a String is a palindrome in Java?**

```
public class PalindromeCheck {
    public static boolean isPalindrome(String str) {
        String reversed = new StringBuilder(str).reverse().toString();
        return str.equals(reversed);
    }

    public static void main(String[] args) {
        System.out.println(isPalindrome("racecar")); // true
        System.out.println(isPalindrome("hello")); // false
    }
}
```

---

**44: How would you identify and count the occurrences of each character in a String?**

```
import java.util.HashMap;

public class CharacterCount {
    public static void countOccurrences(String str) {
        HashMap<Character, Integer> charCount = new HashMap<>();
        for (char c : str.toCharArray()) {
            charCount.put(c, charCount.getOrDefault(c, 0) + 1);
        }

        charCount.forEach((key, value) -> System.out.println(key + ": " + value));
    }

    public static void main(String[] args) {
        countOccurrences("hello world");
    }
}
```

---

**45: Write a Java program to reverse a given String without using the built-in reverse method.**

```
public class ReverseWithoutBuiltIn {
    public static String reverse(String str) {
        StringBuilder reversed = new StringBuilder();
        for (int i = str.length() - 1; i >= 0; i--) {
            reversed.append(str.charAt(i));
        }
```

```
        return reversed.toString();
    }

    public static void main(String[] args) {
        System.out.println(reverse("Hello")); // "olleH"
    }
}
```

---

**46: Create a method that checks if a given String is a palindrome (reads the same forwards and backwards).**

```
public class PalindromeMethod {
    public static boolean isPalindrome(String str) {
        int left = 0, right = str.length() - 1;
        while (left < right) {
            if (str.charAt(left) != str.charAt(right)) {
                return false;
            }
            left++;
            right--;
        }
        return true;
    }

    public static void main(String[] args) {
        System.out.println(isPalindrome("racecar")); // true
        System.out.println(isPalindrome("hello")); // false
    }
}
```

---

**47: Write a program that counts the number of vowels and consonants in a given String.**

```
public class VowelConsonantCount {
    public static void countVowelsAndConsonants(String str) {
        int vowels = 0, consonants = 0;
        for (char c : str.toLowerCase().toCharArray()) {
            if ("aeiou".indexOf(c) != -1) {
                vowels++;
            } else if (Character.isLetter(c)) {
                consonants++;
            }
        }
        System.out.println("Vowels: " + vowels + ", Consonants: " + consonants);
    }

    public static void main(String[] args) {
        countVowelsAndConsonants("Hello World");
    }
}
```

---

**48: Implement a method that capitalizes the first letter of each word in a given String.**

```
public class CapitalizeWords {
    public static String capitalize(String str) {
        String[] words = str.split("\\s+");
        StringBuilder result = new StringBuilder();

        for (String word : words) {
            result.append(Character.toUpperCase(word.charAt(0)))
                  .append(word.substring(1)).append(" ");
        }
```

```
        return result.toString().trim();
    }

    public static void main(String[] args) {
        System.out.println(capitalize("hello world")); // "Hello World"
    }
}
```

---

**49: Implement a method that checks if two Strings are anagrams of each other (contain the same characters in a different order).**

```
import java.util.Arrays;

public class AnagramCheck {
    public static boolean areAnagrams(String str1, String str2) {
        char[] arr1 = str1.toCharArray();
        char[] arr2 = str2.toCharArray();
        Arrays.sort(arr1);
        Arrays.sort(arr2);
        return Arrays.equals(arr1, arr2);
    }

    public static void main(String[] args) {
        System.out.println(areAnagrams("listen", "silent")); // true
    }
}
```

---

**50: Write a program to remove duplicate characters from a String while maintaining the original order of characters.**

```
public class RemoveDuplicates {
    public static String removeDuplicates(String str) {
        StringBuilder result = new StringBuilder();
        for (char c : str.toCharArray()) {
            if (result.indexOf(String.valueOf(c)) == -1) {
                result.append(c);
            }
        }
        return result.toString();
    }

    public static void main(String[] args) {
        System.out.println(removeDuplicates("programming")); // "progamin"
    }
}
```

---

**51. Find the first non-repeating character in a String**

```
public class Main {
    public static char firstNonRepeating(String str) {
        for (char c : str.toCharArray()) {
            if (str.indexOf(c) == str.lastIndexOf(c)) {
                return c;
            }
        }
        return '\0'; // return null character if no non-repeating character
    }
    public static void main(String[] args) {
        System.out.println(firstNonRepeating("swiss")); // Output: w
    }}
```

**52. Compress a String using the counts of repeated characters**

```
public class Main {
   public static String compressString(String str) {
      StringBuilder sb = new StringBuilder();
      int count = 1;

      for (int i = 1; i < str.length(); i++) {
         if (str.charAt(i) == str.charAt(i - 1)) {
            count++;
         } else {
            sb.append(str.charAt(i - 1)).append(count);
            count = 1;
         }
      }
      sb.append(str.charAt(str.length() - 1)).append(count);

      return sb.length() < str.length() ? sb.toString() : str;
   }
   public static void main(String[] args) {
      System.out.println(compressString("aabccccccaaa")); // Output: a2b1c5a3
   }
}
```

---

**53. Append " World" to an existing StringBuffer containing "Hello"**

```
public class Main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("Hello");
      sb.append(" World");
      System.out.println(sb); // Output: Hello World
   }
}
```

---

**54. Insert "Beautiful " at index 6 in "Hello World"**

```
public class Main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("Hello World");
      sb.insert(6, "Beautiful ");
      System.out.println(sb); // Output: Hello Beautiful World
   }
}
```

---

**55. Reverse the contents of a StringBuffer initialized with "Java Programming"**

```
public class Main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("Java Programming");
      sb.reverse();
      System.out.println(sb); // Output: gnimmargorPavaJ
   }
}
```

---

**56. Delete a substring from a StringBuffer**

```
public class Main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("Hello World");
      sb.delete(sb.indexOf("World"), sb.indexOf("World") + "World".length());
      System.out.println(sb); // Output: Hello
   }}
```

**57. Reverse the content of a StringBuffer initialized with "Java Programming"**

```
public class Main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("Java Programming");
      sb.reverse();
      System.out.println(sb); // Output: gnimmargorP avaJ
   }
}
```

---

**58. Delete the substring "World" from "Hello World" and print the modified StringBuffer**

```
public class Main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("Hello World");
      sb.delete(sb.indexOf("World"), sb.indexOf("World") + "World".length());
      System.out.println(sb); // Output: Hello
   }
}
```

---

**59. Replace "Java" with "Python" in "I love Java programming"**

```
public class Main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("I love Java programming");
      int start = sb.indexOf("Java");
      sb.replace(start, start + "Java".length(), "Python");
      System.out.println(sb); // Output: I love Python programming
   }
}
```

---

**60. Check the initial capacity of a StringBuffer and append enough characters to exceed that capacity**

```
public class Main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer();
      System.out.println("Initial capacity: " + sb.capacity()); // Output: Initial capacity: 16
      sb.append("abcdefghijklmnopqrstuvwxyz");
      System.out.println("New capacity: " + sb.capacity()); // Output: New capacity: 34
   }
}
```

---

**61. Write a method that converts a StringBuffer to a String and returns it. Initialize a StringBuffer with "Hello World" and use your method to print the resulting string.**

Input: StringBuffer initialized with "Hello World"
Output: "Hello World"

```java
public class Main {
   public static String convertToString(StringBuffer sb) {
      return sb.toString();
   }

   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("Hello World");
      System.out.println(convertToString(sb)); // Output: Hello World
   }
}
```

---

**62. Create a method that counts the number of vowels in a StringBuffer. Initialize it with any string and print the number of vowels.**
Input: "Hello World"
Output: 3

```
public class Main {
   public static int countVowels(StringBuffer sb) {
      int count = 0;
      for (int i = 0; i < sb.length(); i++) {
         char c = sb.charAt(i);
         if ("AEIOUaeiou".indexOf(c) != -1) {
            count++;
         }
      }
      return count;
   }

   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer("Hello World");
      System.out.println(countVowels(sb)); // Output: 3
   }
}
```

---

**63. Write a Java program that initializes a StringBuffer with extra spaces (e.g., " Hello World ") and trims the whitespace from both ends.**
Input: " Hello World "
Output: "Hello World"

```java
public class Main {
   public static void main(String[] args) {
      StringBuffer sb = new StringBuffer(" Hello World ");
      sb = new StringBuffer(sb.toString().trim());
      System.out.println(sb); // Output: Hello World
   }
}
```

---

**64. Create a method that takes two StringBuffer objects and merges them into one, separating them with a space. Print the resulting StringBuffer.**
Input: StringBuffer1: "Hello", StringBuffer2: "World"
Output: "Hello World"

```java
public class Main {
   public static StringBuffer mergeBuffers(StringBuffer sb1, StringBuffer sb2) {
      return sb1.append(" ").append(sb2);
   }

   public static void main(String[] args) {
      StringBuffer sb1 = new StringBuffer("Hello");
      StringBuffer sb2 = new StringBuffer("World");
      System.out.println(mergeBuffers(sb1, sb2)); // Output: Hello World
   }
}
```